Audio Classification using Deep Learning

Dive into the world of deep learning models for audio classification, where we explore the power of advanced techniques to unlock the secrets hidden within sound waves.

HK

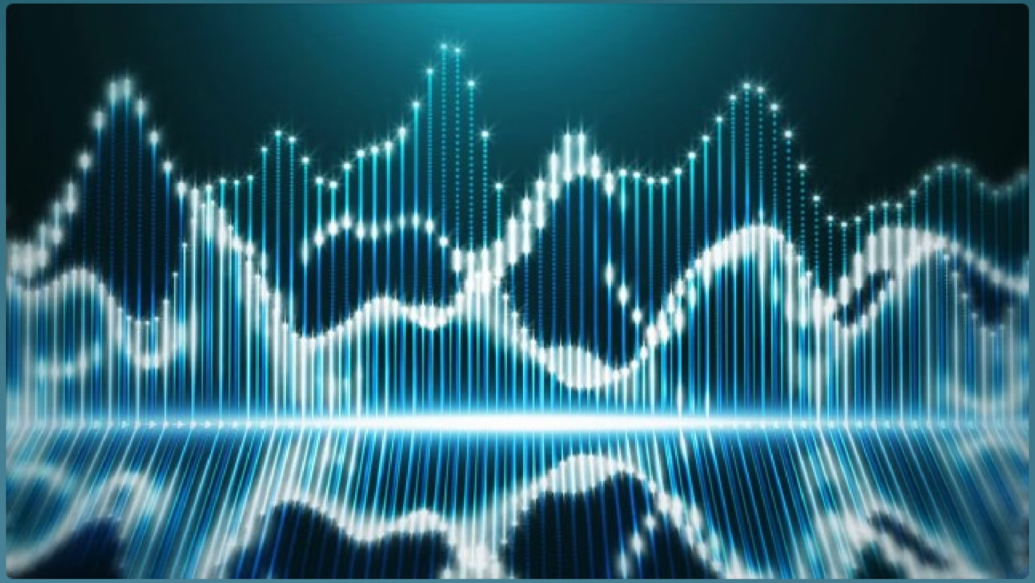
Submitted by :
Harsh Kumar
Aman Pandey
Somnath Mukherjee

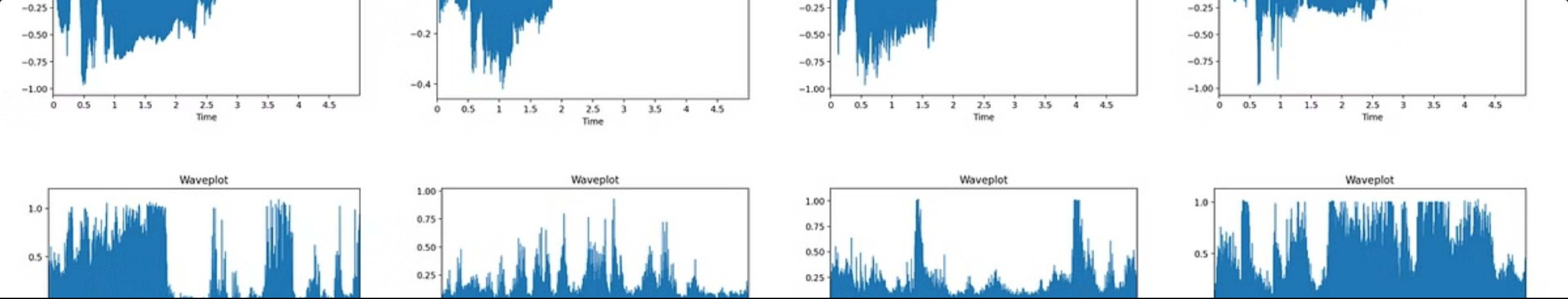
Introduction to Audio Classification and its significance

Audio Classification is a branch of machine learning and signal processing that involves categorizing audio signals into predefined classes or categories. This process leverages algorithm and models to analyze audio features, identifying patterns and characteristics using sound data. These Models are trained on data set containing labeled audio samples, enabling them to learn and predict the class of new, unseen audio input.

The objective of our project is to develop a robust and accurate audio classification system using Deep learning Techniques.

Introduction to Audio Classification and its Significance





Exploratory Data Analysis (EDA)

1

Understanding the Data

Analyze the dataset to gain insights into the types of audio samples, their characteristics, and any potential biases or imbalances.

2

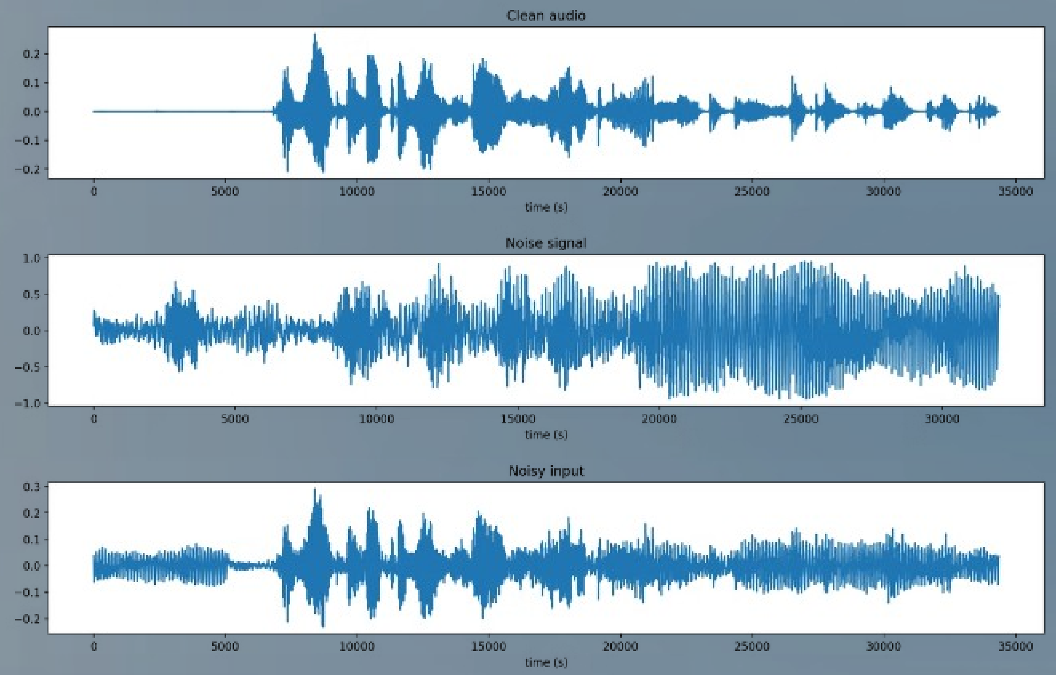
Identifying Patterns

Uncover hidden trends and relationships within the data through statistical analysis and visualization techniques.

3

Informing Preprocessing

Use EDA findings to guide the data preprocessing stage, ensuring the dataset is ready for effective model training.



Data Preprocessing

1

Audio Segmentation

Split the audio samples into manageable segments to facilitate efficient processing and analysis.

2

Feature Extraction

Extract relevant audio features, such as spectrograms, mel-frequency cepstral coefficients (MFCCs), and other signal processing techniques.

3

Normalization

Ensure the data is on a consistent scale, improving the model's ability to converge and perform well.

Model Creation

Neural Network Architecture

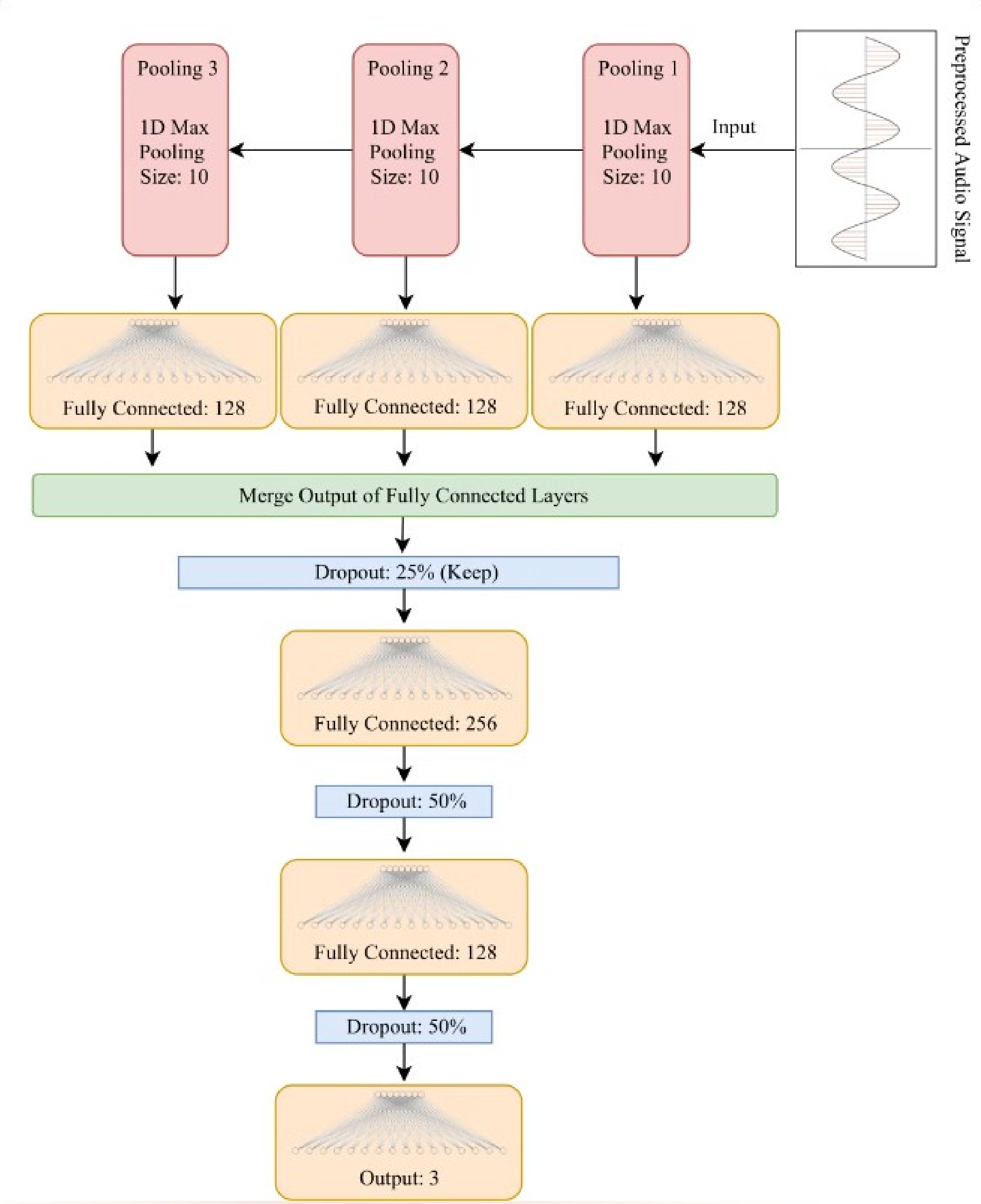
Leverage the power of deep learning models, such as Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs), to capture the intricate patterns in audio data.

Hyperparameter Tuning

Optimize the model's hyperparameters, including learning rate, batch size, and network depth, to achieve the best performance on the task at hand.

Transfer Learning

Explore the potential of transfer learning, where a pre-trained model is fine-tuned on the specific audio classification problem, potentially boosting performance.



Testing the ANN Model

1

Evaluation Metrics

Assess the model's performance using appropriate metrics, such as accuracy, precision, recall, and F1-score, to understand its strengths and limitations.

2

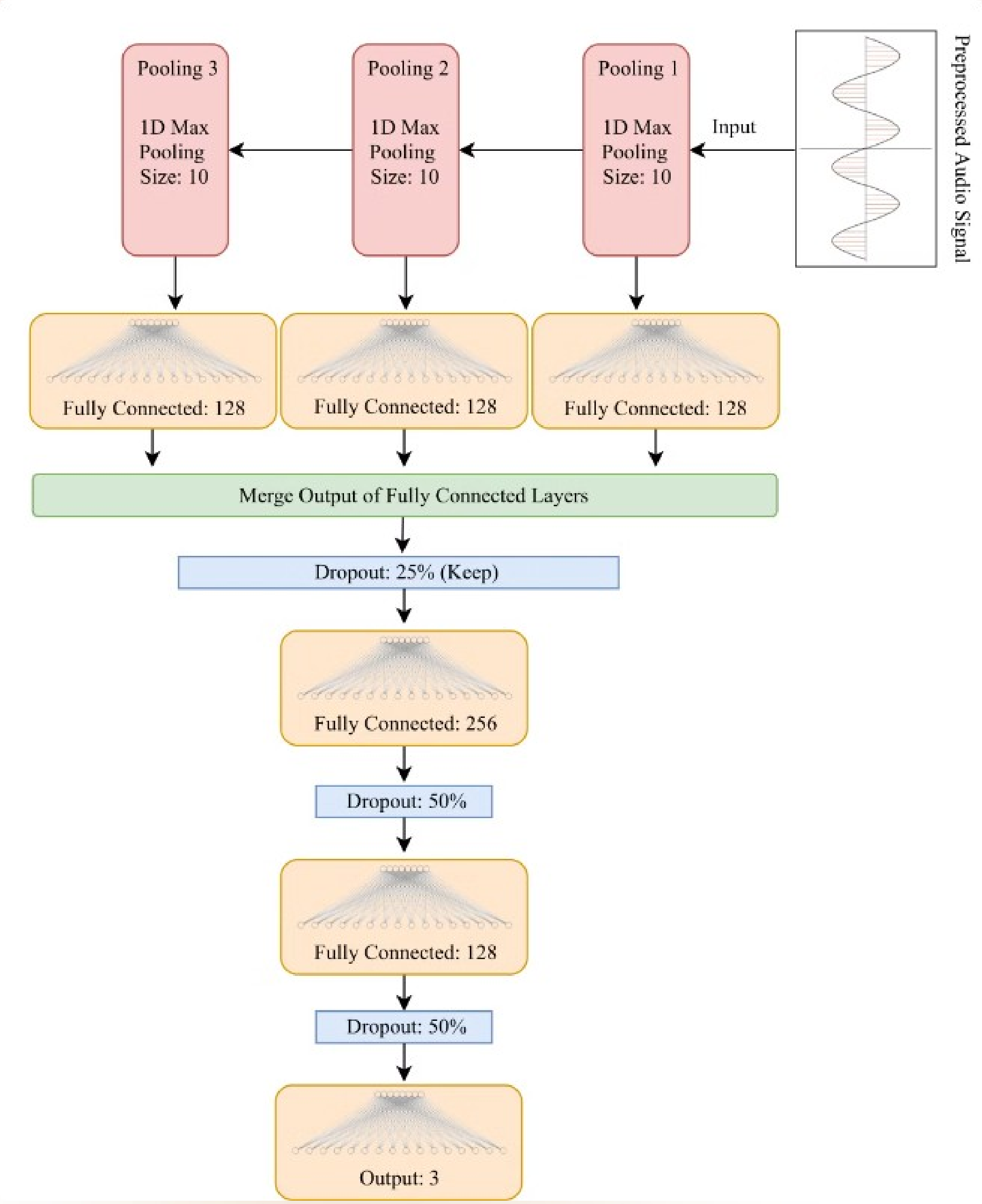
Confusion Matrix

Analyze the confusion matrix to identify any misclassifications and understand the model's ability to distinguish between different audio classes.

3

Robustness Testing

Evaluate the model's resilience to noisy or corrupted audio samples, ensuring it can handle real-world scenarios effectively.



Testing the CNN Model

1

Evaluation Metrics

Assess the model's performance using appropriate metrics, such as accuracy, precision, recall, and F1-score, to understand its strengths and limitations.

2

Robustness Testing

Evaluate the model's resilience to noisy or corrupted audio samples, ensuring it can handle real-world scenarios effectively.

Code Walkthrough

Data Preprocessing

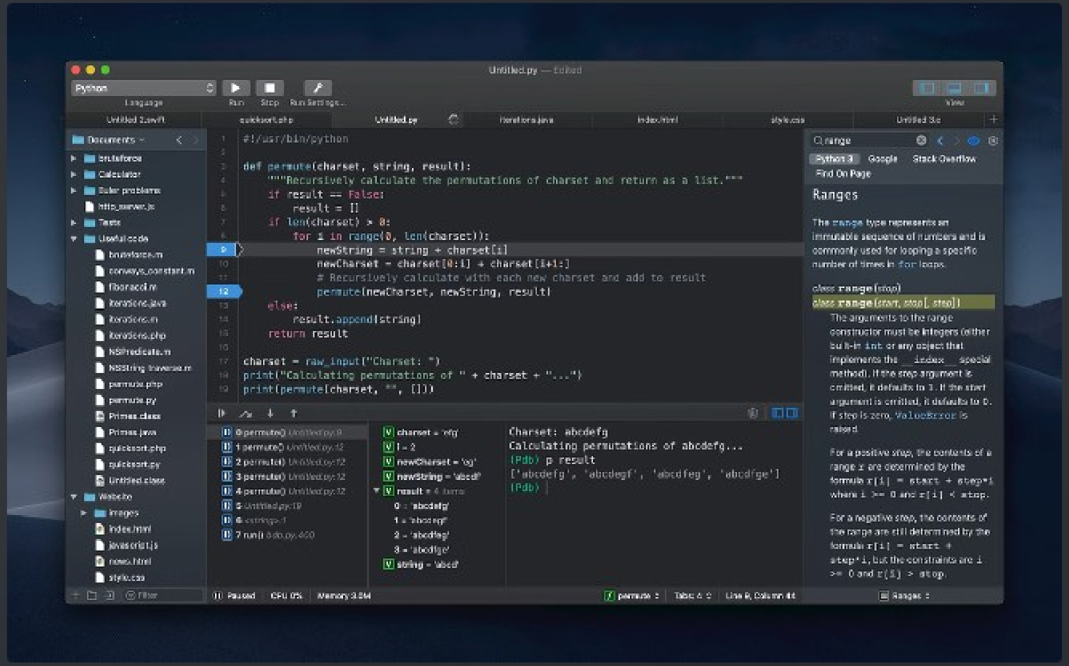
Showcase the code for audio segmentation, feature extraction, and normalization.

Model Architecture

Demonstrate the implementation of the deep learning model, including the neural network layers and hyperparameter tuning.

Model Evaluation

Highlight the code for evaluating the model's performance, including the calculation of metrics and the generation of the confusion matrix.



```python
#!/usr/bin/python

def permute(charset, string, result):
    """Recursively calculate the permutations of charset and return as a list."""
    if result == False:
        result = []
    if len(charset) > 0:
        for i in range(0, len(charset)):
            newString = string + charset[i]
            newCharset = charset[0:i] + charset[i+1:]
            # Recursively calculate with each new charset and add to result
            permute(newCharset, newString, result)
    else:
        result.append(string)
    return result

charset = raw_input("Charset: ")
print("Calculating permutations of " + charset + "...")
print(permute(charset, "", []))
```

```
0 permute() Untitled.py:9
1 permute() Untitled.py:12
2 permute() Untitled.py:12
3 permute() Untitled.py:12
4 permute() Untitled.py:12
5 Untitled.py:19
6 <string>:1
7 run() bdb.py:400
```

```
charset = 'efg'
i = 2
newCharset = 'eg'
newString = 'abcdf'
result = 4 items
  0 = 'abcdefg'
  1 = 'abcdegf'
  2 = 'abcdfeg'
  3 = 'abcdfge'
string = 'abcd'
```

```
Charset: abcdefg
Calculating permutations of abcdefg...
(Pdb) p result
['abcdefg', 'abcdegf', 'abcdfeg', 'abcdfge']
(Pdb)
```

range

Python 3 Google Stack Overflow Find On Page

Ranges

The range type represents an immutable sequence of numbers and is commonly used for looping a specific number of times in for loops.

class range(stop)
class range(start, stop[, step])

The arguments to the range constructor must be integers (either built-in int or any object that implements the __index__ special method). If the step argument is omitted, it defaults to 1. If the start argument is omitted, it defaults to 0. If step is zero, ValueError is raised.

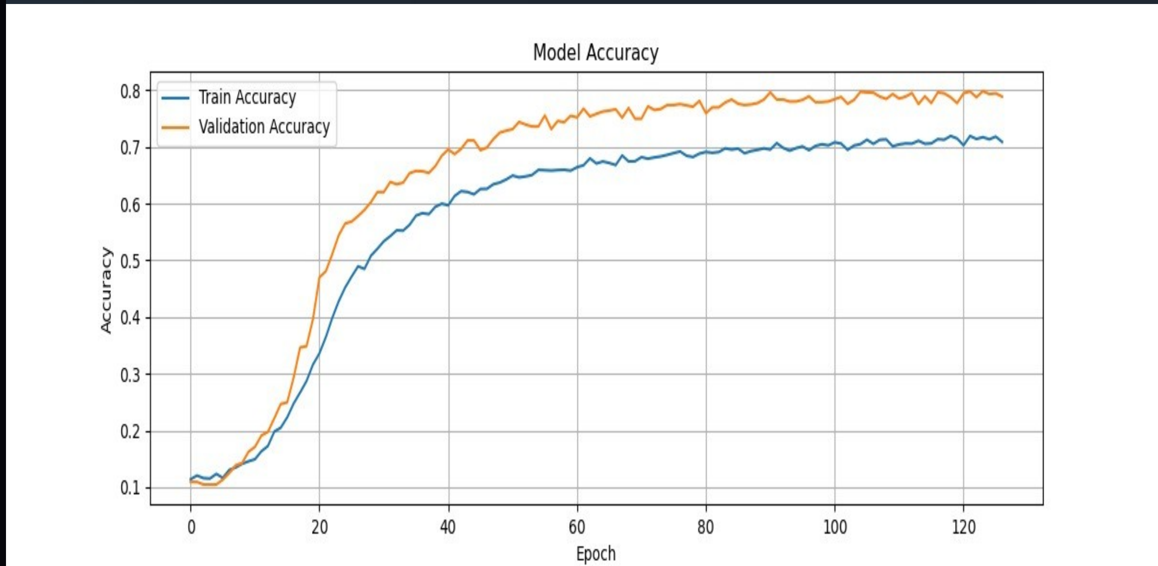
For a positive step, the contents of a range r are determined by the formula r[i] = start + step*i where i >= 0 and r[i] < stop.

For a negative step, the contents of the range are still determined by the formula r[i] = start + step*i, but the constraints are i >= 0 and r[i] > stop.

Conclusion

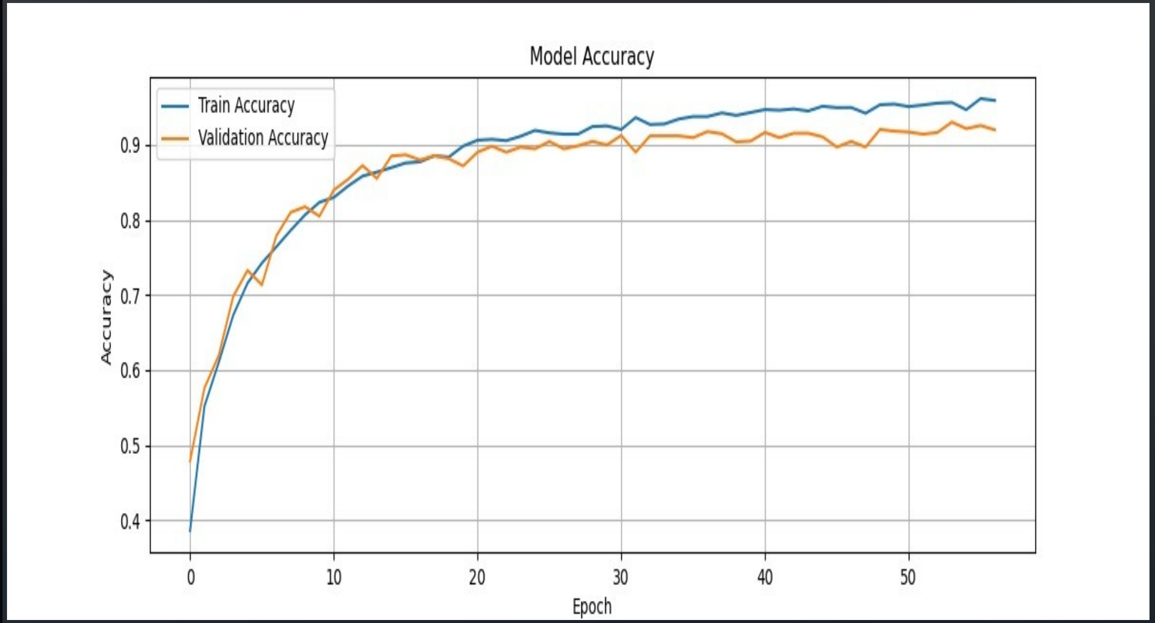
ANN Model Evaluation Metrics

Achieved 80% Accuracy



CNN Model Evaluation Metrics

Achieved 92% Accuracy.



Key Takeaways

Powerful Feature Engineering

The importance of extracting relevant audio features to capture the underlying patterns in the data.

Model Optimization

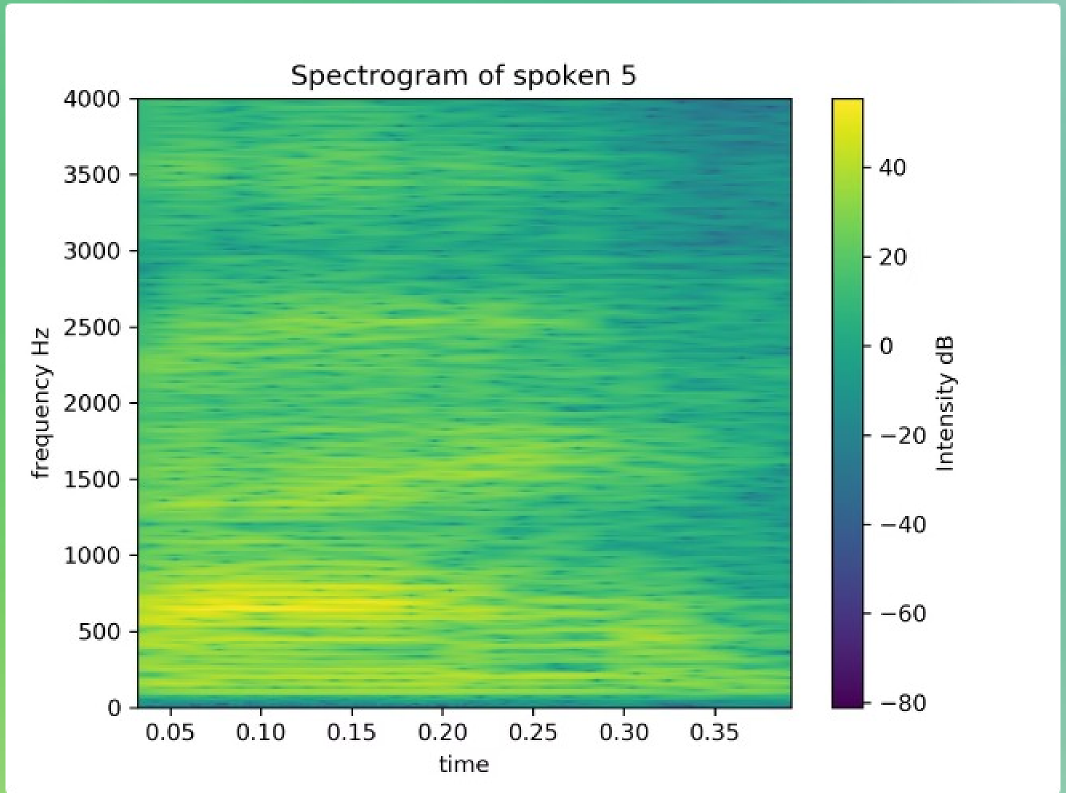
The significance of hyperparameter tuning and the potential benefits of transfer learning for improved model performance.

Robust Evaluation

The need for comprehensive evaluation, including the use of various metrics and techniques like the confusion matrix.

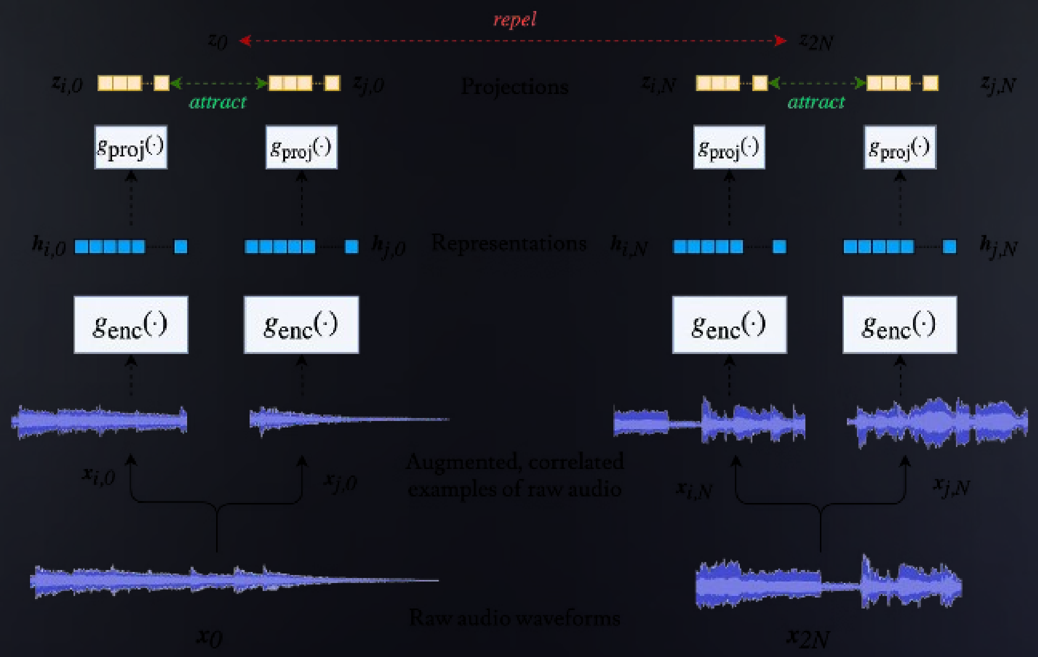
Future Directions

Opportunities for further research and development in the field of audio classification using deep learning.



Thank You

This presentation has provided an in-depth look at the exciting world of audio classification using deep learning. We've explored the key stages of the process, from data analysis to model creation and testing, showcasing the power of these advanced techniques. Thank you for your attention, and let's continue exploring the possibilities of this fascinating field.


repel
z_0
z_{2N}
$z_{i,0}$
attract
$z_{j,0}$
Projections
$z_{i,N}$
attract
$z_{j,N}$
$g_{\text{proj}}(\cdot)$
$g_{\text{proj}}(\cdot)$
$g_{\text{proj}}(\cdot)$
$g_{\text{proj}}(\cdot)$
$h_{i,0}$
$h_{j,0}$
Representations
$h_{i,N}$
$h_{j,N}$
$g_{\text{enc}}(\cdot)$
$g_{\text{enc}}(\cdot)$
$g_{\text{enc}}(\cdot)$
$g_{\text{enc}}(\cdot)$
$x_{i,0}$
$x_{j,0}$
Augmented, correlated examples of raw audio
$x_{i,N}$
$x_{j,N}$
Raw audio waveforms
x_0
x_{2N}